







 **zxjcarrot** Update README.md ...

on Feb 24  9

	include	Initial code drop.	12 months ago
	misc	Initial code drop.	12 months ago
	src	Initial code drop.	12 months ago
	test	Initial code drop.	12 months ago
	CMakeList...	Initial code drop.	12 months ago
	LICENSE.md	Create LICENSE.md	12 months ago
	README.md	Update README.md	4 months ago

 README.md

Source code for Spitfire: A Three-Tier Buffer Manager for Volatile and Non-Volatile Memory

-  Readme
-  MIT license
-  **20** stars
-  **3** watching
-  **9** forks

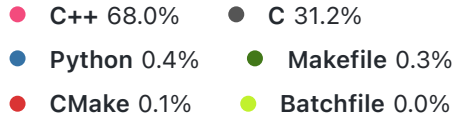
Releases

No releases published

Packages

No packages published

Languages



Spitfire: A Three-Tier Buffer Manager for Volatile and Non-Volatile Memory

A multi-threaded buffer manager built for multi-tier storage hierarchy involving DRAM/NVM/SSD. Check out our [SIGMOD 2021 paper](#) for more details. This repo contains implementations of the ideas and experiments discussed in paper:

- Probabilistic data migration policy.
- Lock-free clock replacement policy.
- Optimistic lock coupling B-tree index built on top of the buffer manager.
- HYMEM optimizations - Cache-lined-grained page, mini page etc.
- YCSB/TPCC benchmark.
- ...

If you use this work, please cite our paper as follows

```
@inproceedings{zhou2021spitfire,  
  title={Spitfire: A Three-Tier Buffer Manager  
for Volatile and Non-Volatile Memory},  
  author={Zhou, Xinjing and Arulraj, Joy and  
Pavlo, Andrew and Cohen, David},  
  booktitle={Proceedings of the 2021  
International Conference on Management of  
Data},  
  pages={2195--2207},  
  year={2021}  
}
```

Building

Dependencies

Spitfire requires `tcmalloc` and Intel's `tbb` library. To install the dependencies on Ubuntu 20.04, run the following command:

```
sudo apt install libgoogle-perftools-dev libtbb-
```

Compilation

```
git clone https://github.com/zxjcarrot/spitfire.  
cd spitfire  
mkdir build  
cd build  
cmake -DCMAKE_BUILD_TYPE=Release ..
```

```
make -j 16 ycsb
```

This generates a `ycsb` executable in build directory.

Playing with Spiftire

Command line options available for the `ycsb` benchmark are as follows:

```
cd build
./ycsb -h
Command line options : ycsb <options>
-h --help                : print help message
-k --scale_factor        : # of K tuples
-d --duration            : execution duration
-p --profile_duration    : profile duration
-b --backend_count       : # of backends
-o --operation_count     : # of operations
-u --update_ratio        : fraction of updates
-z --zipf_theta          : theta to control sk
-e --exp_backoff         : enable exponential
-l --loader_count        : # of loaders
-B --bp_mode            : 0 (DRAM|DRAM|SSD),
-P --nvm_buf_path       : directory in which
-J --wal_path            : logging directory
-D --db_path            : directory where the
-M --mini_page          : enable the mini pag
-I --direct_io          : enable direct io
-Q --dram_read_prob      : probability of a pa
-W --dram_write_prob     : probability of a pa
-E --nvm_read_prob       : probability of a pa
-R --nvm_write_prob      : probability of a pa
-T --dram_buf_num_pages : # pages(16KB) in dr
-Y --nvm_buf_num_pages  : # pages(16KB) in nv
-L --load_existing_db   : whether to load dat
-U --warmup_duration    : warmup duration(s)
-A --enable_annealing   : whether to enable s
-s --shuffle_keys       : whether to shuffle
-t --enable_hyem        : whether to enable H
-X --admission_set_sz   : size of the admissi
```

Hardware Setup for Logging and NVM buffer pool.

Spitfire implements classic redo/undo logging and it optimizes logging by placing the log buffer and the log files on NVM. Spitfire places the NVM buffer on NVM-backed filesystem using `mmap`. Therefore, you need to configure the Optane DIMM in `app-direct` mode and mount an `fsdax` mode file system on top of the device. Check out this [tutorial](#) on how to configure the device and the file system. Once the file system is configured and mounted, create two directories for storing NVM log files and buffer. Then you should pass them to the `ycsb` program using `-J` and `-P` options. Make sure you have the permission to read and write to the files in these directories.

Buffer Pool Mode

The buffer pool can be configured into 4 modes (2 three-tier modes and 2 two-tier modes) using `-B` or `--bp_mode` options.

DRAM-NVM-SSD mode

In this three-tier mode, real NVM is used for the middle buffer. This mode requires `-P` option. For example,

```
./ycsb -k 512 -T 7480 -Y 14960 -I -U 30 -s -J /mnt/myPMem/spitfire_wal/ -P /mnt/myPMem/spitfire_buf/
```

This first loads a 1 GB YCSB database. It writes a nvm log to `/mnt/myPMem/spitfire_wal/`. The NVM buffer is stored in `/mnt/myPMem/spitfire_buf/`. The database files are stored in `/mnt/optane/spitfire_db/`. The DRAM/NVM buffer capacities are set to 116MB and 233MB respectively. After the database is loaded, it warms up the buffers for 60s and then executes YCSB readonly workload for 360s using one thread. You can also use `-L` option to skip the loading process and reload an existing database.

DRAM-NVM(DRAM-emulated)-SSD mode.

In this three-tier mode, the middle buffer emulates NVM using DRAM.

```
./ycsb -k 512 -T 7480 -Y 14960 -I -U 30 -s -J /mnt/myPMem/spitfire_wal/
```

This is similar to the first mode except that the `-P` option is omitted as it does not use NVM.

DRAM-SSD mode

This is a classic 2-tier configuration where DRAM buffer is used to cache a set of hot pages. For example,

```
./ycsb -k 512 -T 7480 -I -U 30 -s -J /mnt/myPMem
```

This uses a 233MB of DRAM buffer on top of the database.

NVM-SSD mode

In this two-tier mode, a persistent NVM buffer is used to cache hot pages. Different from DRAM-SSD mode, writes to pages in NVM buffer are persistent and requires no dirty page flush except for evictions. For example,

```
./ycsb -k 512 -Y 14960 -I -U 30 -s -J /mnt/myPMem
```

This uses a 233MB of NVM buffer on top of the database. The NVM buffer is stored in `/mnt/myPMem/spitfire_buf/`.

Lazy Migration Policy

By default, Spitfire uses Eager migration policy. You can use `-Q`, `-W`, `-E`, `-R` to adjust the migration probabilities regarding DRAM and NVM. For example

```
./ycsb -k 512 -T 7480 -Y 14960 -I -U 30 -s -J /
```

This configuration uses the Lazy migration policy for DRAM-NVM-SSD mode described in the paper.

Adaptive Data Migration

To try out simulated-annealing, use `-A` option.

```
./ycsb -k 512 -T 1496 -Y 14960 -I -U 30 -s -J /
```

This instructs Spitfire to start from an eager policy and gradually tune the policy over the duration of the benchmark.

HYMEM optimizations

Spitfire codebase includes two HYMEM optimizations:

- Fine-grained Page Loading
- Mini Page

For the first optimization, the loading granularity is defined in `include/config.h`. By default, the loading granularity is 16384 which is a full page. You can change this number and recompile to enable this optimization.

For the Mini Page optimization, you can use the `-M` knob.

Caveats

- The code is only tested on Ubuntu 20.04 with gcc 9.3.0 toolchains.
- The recovery protocol is not fully implemented.